Express Mail Mailing Label No. EL 988705055 US

PATENT APPLICATION
Atty. Docket No.: SNS-016 (7268/24)

APPARATUS AND METHODS FOR STENCILING AN IMAGE

Related Applications

[0001] This application is related to the commonly-owned U.S. patent application entitled, "Apparatus and Methods for Texture Mapping," by Levene et al., filed under Attorney Docket No. SNS-015, on even date herewith, the text of which is hereby incorporated by reference in its entirety.

Field of the Invention

[0002] This invention relates generally to computer graphics applications. More particularly, in certain embodiments, the invention relates to image editing functions in computer graphics applications.

Background of the Invention

[0003] Certain computer graphics applications include a stenciling function for protecting part of an image from subsequent paint strokes. A stenciling function allows a user to select a region of the image to protect, such that subsequent paint strokes falling within the selected region are not blended into the image.

[0004] A stenciling function may also allow a user to apply a *partial* mask to a selected region, such that a subsequent paint stroke falling within the selected region is only

partially blended into the image. For example, a partial mask may be used along the edges of a full mask in order to provide a smooth transition between masked and unmasked portions of the stenciled image.

[0005] A partial mask may be applied for a single paint stroke falling within a selected region by reducing the intensity of the paint stroke prior to blending the paint stroke into the image. Partial masks that are available in current graphics applications are of limited value where paint strokes overlap. Overlapping paint strokes within a partially masked region may result in a blending of more color into the underlying image than is desired.

[0006] Paint strokes within a partially stenciled region are blended into an image one by one, as soon as each paint stroke is executed. The color contributed by the original layer appears to diminish as new layers of paint are blended into the partially-stenciled region. That is, the opacity of the original layer decreases as the image is layered over with new paint.

[0007] It is not possible for a user to specify that an initial version of a selected region of an image be maintained at a specified opacity in any subsequent composite. This is because the image is modified after each stroke in order to maintain real-time interactivity with the user, and previous brush strokes are not accounted for in the attenuation and blending of subsequent, overlapping strokes.

[0008] Similarly, overlapping erase strokes within a partially-masked region may result in a removal of more color than is desired by the user, since overlapping erase strokes in the selected region will eventually reduce the opacity of the original layer below the desired minimum.

[0009] Furthermore, current methods of blending individual paint strokes into an image may result in undesired artifacts. An individual paint stroke is generally divided into a series of stroke segments that are separately blended into an image. Overlapping portions at the ends of segments of an individual paint stroke are blended twice, causing undesired artifacts.

[0010] There exists a general need for more accurate and more robust stenciling methods in computer graphics applications. More specifically, there exists a need for stenciling methods that allow a user to specify that a selected region not change more than a specified amount during subsequent editing.

Summary of the Invention

[0011] The invention provides methods for protecting a selected region of an image from subsequent editing. More specifically, the invention provides stenciling methods that allow a user to more accurately control the editing of an image by specifying a maximum amount by which selected portions of the image are allowed to change over the course of a series of brush strokes.

[0012] The methods involve protecting an image using a first texture and a second texture, rather than a single texture alone. The first texture includes pixels with values indicating levels of protection to be applied to corresponding pixels of the protected image. For example, the first texture may be a map of values indicating one or more regions of the image to be protected from editing by subsequent paint strokes. The level of protection may be from 0% to 100%, where 0% indicates no protection from subsequent edits, and 100% indicates full protection from subsequent edits. A level

between 0% and 100% indicates partial protection, and may correspond to a minimum opacity above which to maintain at least a portion of the protected region throughout the application of subsequent brush strokes.

[0013] Graphical input is directed into a second texture, rather than being directly blended into the protected image. In a preferred embodiment, the second texture accumulates graphical input as long as the stencil remains active. The accumulated graphical input may represent one or more brush strokes performed by a user. Values of the second texture are modified using corresponding first texture values, and are blended into the protected image. The modification and blending of the accumulated data may be performed substantially at the same time, as in a consolidated operation.

[0014] Because the second texture is able to accumulate graphical data from more than one brush stroke, methods of the invention allow a user to maintain an initial version, or initial layer, of a selected region of an image at a specified opacity in any subsequent composite, regardless of overlapping brush strokes within the selected region. This is done by accumulating graphical input representing one or more brush strokes in the second texture, and by *subsequently* blending the pixels of the second texture, modified by first texture values, into the protected image. The blending is performed, for example, using a compositing function.

[0015] Overlapping portions may result from multiple overlapping brush strokes and/or from a single brush stroke that overlaps itself. Despite the presence of any overlapping portion(s), and despite the number of brush strokes applied following activation of the stencil, methods of the invention can prevent the opacity of an initial version, or initial

layer, of the selected region(s) from decreasing below a specified minimum in any subsequent composite.

[0016] The interactivity of the user's image editing experience may be preserved by use of a display image. In one embodiment, pixels of the protected image are copied directly into a display image, while pixels of the second texture are modified according to the first texture and blended into the display image. The modified pixels of the second texture may be blended into the display image, for example, by compositing the second texture over the display image.

[0017] User interactivity is preserved by modifying pixels of the second texture and blending the modified pixels into the display image on a pixel-by-pixel basis. In this way, the user sees the resulting image emerge, subject to the user-specified protection, as the user continues to apply brush strokes.

[0018] The display image can reflect real-time user brush strokes, as well as preserve a minimum opacity of the original image layer within a protected region, regardless of the number of brush strokes that follow. This is because, in a preferred embodiment, each update of the display image is performed by: (1) re-copying pixel values of the original protected image layer into the display image pixels, and then (2) compositing the modified second texture pixels with the display image pixels. The display image may be updated at a rate of up to about 30 times per second, or more, depending on the complexity of the image and the computational speed for graphical rendering. The update rate may be less for more complex images or for slower machines – for example, from about 10 times per second to about 20 times per second.

[0019] The use of a display image is optional. Whether or not a display image is used, methods of the invention can preserve a minimum opacity of the original image layer within a protected region by accumulating graphical input in the second texture, and by subsequently blending pixels of the second texture, modified by first texture values, into the protected image. The protected image is not affected by the accumulated graphical input until the final blending step. Thus, in one embodiment, after all the graphical input representing a plurality of brush strokes are accumulated and modified, the second texture is blended into the protected original image.

[0020] The user may provide one or more signals indicating the beginning and/or ending of the period of time in which brush strokes are accumulated in the second texture. The user provides a first signal, such as a button click, to indicate the beginning of the application of a stencil. Alternatively, the stencil may self-activate once the user defines the first texture. The step of defining the first texture may include indicating: (1) one or more regions of an image to be protected; and/or (2) one or more levels of protection to apply within the one or more regions.

[0021] Once the stencil is active, graphical input representing brush strokes by the user are directed into the second texture. When the user has completed one or more brush strokes, the user may provide a second signal to deactivate the stencil. In one embodiment, once the stencil is deactivated, the second texture, modified by the first texture, is blended into the protected image.

[0022] Real-time display of erase strokes requires modification of paint strokes applied both before and after activation of the stencil. Generally, the second texture contains data from paint strokes applied after activation of the stencil, but not before. Therefore, one

embodiment of the invention includes the step of modifying a value of one or more pixels of the protected image *prior to* copying pixels of the protected image into the display image and compositing the modified second texture pixels with the display image pixels. Pixel values of the protected image are modified according to the level(s) of protection indicated in the first texture.

[0023] In one embodiment, a minimum opacity of an original image layer may be preserved despite repeated erase strokes within the protected region, where overlapping erase strokes result in the attenuation of a pixel value of the protected image down to a minimum value, but no further. The minimum value is determined from the protection level of that pixel as indicated in the first texture. In cases where pixels are represented by RGBA quadruples, the first texture may be used to derive a minimum alpha channel value below which the pixel values of the protected image are not allowed to fall.

[0024] Thus, the display methods discussed herein work where a user performs paint strokes, erase strokes, or both paint and erase strokes, while the stencil remains active. Additionally, the display methods preserve the interactivity experienced by the user during the editing process, while maintaining the protection offered by the stenciling methods described herein.

[0025] Methods of the invention involve the modification of paint colors of images and/or textures, as well as the blending of paint colors between two or more images and/or textures. For example, in one embodiment, pixels of the second texture are modified using the first texture, where the first texture is a map of values indicating protection levels for the image and the second texture contains accumulated graphical

input. A pixel of the second texture may be modified by attenuating its color value according to the protection level in the first texture corresponding to that pixel.

[0026] Blending of images and/or textures may be performed using a compositing operation. For example, in one embodiment, pixels of the second texture are blended into the protected image. This may be done by performing a compositing operation, such as an overlay operation in RGBA colorspace, between the modified values of the second texture and the protected image.

[0027] In addition to the stenciling methods summarized above, the invention provides a blending method that prevents double-blending of overlapping portions of segments of an individual brush stroke, thereby avoiding undesired artifacts at the segment ends. An individual brush stroke is typically divided into a series of stroke segments that are separately blended into a target image. For example, each segment of an individual brush stroke may be represented with a pillbox shape. The blending method avoids higher intensities in the circular regions where the ends of the pillboxes overlap.

[0028] The blending method uses a scratch texture to describe brush characteristics, for example, the scratch texture may contain intensity values that reflect brush size and/or edge effects. By assigning a new pixel value to the scratch texture only if it exceeds the existing value at that pixel, the double-blending artifact is avoided. In one embodiment, the blending method includes the steps of: (1) receiving brush stroke data from a graphical user interface; (2) for each of a plurality of pixels of a scratch texture, comparing a received pixel value to a value previously written at the corresponding pixel of the scratch texture and assigning the received value to the corresponding pixel of the scratch texture only if it exceeds the existing value; and (3) blending the scratch texture

into the target image. In one embodiment, the blending step is performed substantially upon completion of the performance of the paint stroke by the user.

[0029] The blending method is compatible with use of a scratch texture for representing a transition region along one or more edges of a brush stroke. The "falloff" of a brush stroke at the edges of the brush may be captured in the scratch texture. Here, a pixel value in the transition region may be determined as a function of a distance of the pixel from the center of the brush stroke.

[0030] The use of a scratch texture to characterize an individual brush stroke is compatible with the stenciling methods summarized above. For example, the graphical input used in the stenciling methods summarized above may include a scratch texture representing a brush stroke. The stenciling method may include blending the scratch texture into the second texture, wherein the second texture accumulates one or more brush strokes performed while the stencil is active. The brush stroke may include a falloff region, which is accounted for in the scratch texture. In one embodiment, the scratch texture includes intensity values which correspond to a transition region at the edges of the brush stroke.

Brief Description of the Drawings

[0031] The objects and features of the invention can be better understood with reference to the drawings described below, and the claims. The drawings are not necessarily to scale, emphasis instead generally being placed upon illustrating the principles of the invention. In the drawings, like numerals are used to indicate like parts throughout the various views. The patent or application file contains at least one drawing executed in color. Copies of this patent or patent application publication with color drawing(s) will be provided by the U.S. Patent and Trademark Office upon request and payment of the necessary fee.

[0032] Figure 1 is a block diagram featuring a method of blending paint strokes and erase strokes into an image prior to activation of a stencil or following deactivation of a stencil, according to an illustrative embodiment of the invention.

[0033] Figure 2 is a schematic diagram illustrating a method for blending graphical input into an image while wholly or partially protecting one or more selected regions of the image from paint edits, according to an illustrative embodiment of the invention.

[0034] Figure 3 is a block diagram featuring the stenciling method illustrated in Figure 2, according to an illustrative embodiment of the invention.

[0035] Figure 4A is a schematic diagram illustrating the result of applying orange paint strokes over a circular blue stroke without a stencil.

[0036] Figure 4B is a schematic diagram illustrating the result of applying orange paint strokes over a circular blue stroke following activation of a 100% protection level stencil, according to an illustrative embodiment of the invention.

[0037] Figure 5 is a schematic diagram showing pixel values of a scratch texture as grayscale intensity levels, where the scratch texture represents a paint stroke with a transition region, according to an illustrative embodiment of the invention.

[0038] Figure 6 shows a graph depicting the intensity levels of an illustrative brush stroke as a function of a distance from the center of the brush stroke, where the brush stroke has a transition region, and the intensity levels are contained in a scratch texture for the brush stroke, according to an illustrative embodiment of the invention.

[0039] Figure 7A is a block diagram featuring a method of blending graphical input into a target image without using a scratch texture, according to an illustrative embodiment of the invention.

[0040] Figure 7B is a block diagram featuring a method of blending graphical input into a target image using a scratch texture, according to an illustrative embodiment of the invention.

[0041] Figure 8 is a schematic diagram demonstrating a buildup of color due to multiple overlapping paint strokes within a partially-masked region.

[0042] Figure 9 is a schematic diagram demonstrating how an illustrative stenciling method of the invention prevents buildup of color due to multiple overlapping paint strokes within a partially-masked region.

[0043] Figure 10A is a schematic diagram illustrating a circular paint stroke having a blue color value, according to an illustrative embodiment of the invention.

[0044] Figure 10B is a schematic diagram illustrating the result of applying two overlapping orange paint strokes, each at 50% intensity, onto the circular blue paint

stroke of Figure 10A, without a stencil, according to an illustrative embodiment of the invention.

[0045] Figure 10C is a schematic diagram illustrating the result of applying two overlapping orange paint strokes, each at 50% intensity, onto the circular blue paint stroke of Figure 10A after activation of a 50% protection stencil over the entire circular region, according to an illustrative embodiment of the invention.

[0046] Figure 10D is a schematic diagram illustrating the result of applying three overlapping orange paint strokes, each at 50% intensity, onto the circular blue brush stroke of Figure 10A after activation of a 50% protection stencil over the entire circular region, according to an illustrative embodiment of the invention.

[0047] Figure 11 is a schematic diagram illustrating a method for blending erase strokes into an image while wholly or partially protecting one or more selected regions of the image from edits, according to an illustrative embodiment of the invention.

[0048] Figure 12 is a block diagram featuring the stenciling method illustrated in Figure 11, according to an illustrative embodiment of the invention.

[0049] Figure 13 is a schematic diagram demonstrating a removal of color due to multiple overlapping erase strokes within a partially-masked region.

[0050] Figure 14 is a schematic diagram demonstrating how an illustrative stenciling method of the invention prevents removal of color due to multiple overlapping erase strokes within a partially-masked region.

[0051] Figure 15 is a schematic diagram illustrating the result of applying two overlapping erase strokes, each at full intensity, onto a circular blue paint stroke after

activation of a 50% protection stencil over the entire circular region, according to an illustrative embodiment of the invention.

[0052] Figure 16A is a schematic diagram illustrating the application of a single paint stroke across an image following activation of a partial mask along the edges of a full mask, where the partial mask provides a smooth transition between masked and unmasked portions of the stenciled image.

[0053] Figure 16B is a schematic diagram illustrating the buildup of color within the partially protected region of the image of Figure 16A following application of three overlapping paint strokes across the entire image.

[0054] Figure 16C is a schematic diagram illustrating the further buildup of color within the partially protected region of the image of Figure 16A following application of five overlapping paint strokes across the entire image.

[0055] Figure 17 is a schematic diagram showing two contiguous pillbox segments of a brush stroke both with and without a double-blending artifact in the overlapping portion, according to an illustrative embodiment of the invention.

[0056] Figure 18 is a block diagram featuring a method of blending a scratch texture containing a single brush stroke into a target image, according to an illustrative embodiment of the invention.

Detailed Description

[0057] In general, the invention relates to methods for protecting a selected region of an image from subsequent editing. More specifically, the invention provides methods of specifying a maximum amount by which one or more selected portions of an image are allowed to change during the course of a series of brush strokes.

[0058] As used herein, a brush stroke is an operation performed on one or more pixels of an image. Brush strokes include, for example, paint strokes, erase strokes, pencil strokes, pen strokes, lines, characters, and text.

[0059] Each brush stroke is divided into segments that link positions of a graphical interface device, such as a stylus or other painting tool. The positions are recorded at successive frames as the user moves the graphical user interface device in real space, and, preferably, while the user views a graphical rendering of the image upon which the user is applying the brush stroke. Each brush stroke may be represented as having a skeleton made up of a series of straight line segments connecting the recorded positions of the graphical user interface device. Each individual segment of a brush stroke may be represented, for example, by a pillbox having a specified width about the stroke skeleton and having curved, semicircular ends. Alternately, each segment may be represented by another shape having a fixed or variable width along the length of the stroke segment and having a given shape at the end of the stroke segment.

[0060] A user may interactively determine the manner in which a brush stroke operates on a collection of pixels of an image, as in the performance of a paint or erase stroke, using a graphical interface device. Alternatively, application of a brush stroke may be largely non-interactive, executed with minimal user input. Thus, brush strokes may

include, for example, batch deletions, batch pastes, and flood fills. The pixels on which a brush stroke operates may be either contiguous or non-contiguous. The graphical interface device may be a two dimensional input device, such as a mouse, a pen, a graphics tablet. Alternatively, the graphical interface device may be a 3, 4, 5, 6, or higher-degree-of-freedom device. The user may use the graphical interface device to apply brush strokes onto a two-dimensional surface. The user may also use the graphical interface device to apply brush strokes onto the surface of a three-dimensional virtual object. The graphical interface device may be a haptic interface device, compatible with a system for haptically rendering a virtual object, where the haptic interface device provides force feedback to the user in response to a position of the user within a virtual environment. The virtual object may be rendered both graphically and haptically.

[0061] The invention protects a selected portion of a target image by using two textures, rather than a single texture alone. A texture is an array of values. For example, a texture may be a two-dimensional array of values, wherein each value corresponds to a location within a two-dimensional region. Texture values may define visual attributes of an image, or may represent any other characteristic. For example, a texture may contain values representing levels of protection to be applied to corresponding pixels of a protected image. A texture may be displayed as an image, where texture values provide visual attributes applicable to corresponding positions. A texture may be stored, for example, as a file or other collection of data. The terms texel and pixel are used interchangeably herein. The term texel generally refers to a unit of a texture, and the term pixel generally refers to a unit of either a texture or an image.

[0062] Pixel values and/or texel values include values of any bit depth, where bit depth generally ranges from 1 to 64 bits per pixel. Pixel values include grayspace values, RGB colorspace values, RGBA colorspace values, or any other colorspace values, such as HSL, HSV, CMY, CMYK, CIE Lab, and R-Y/B-Y. Preferred methods of the invention are performed using 24-bit, RGBA colorspace pixels, although any bit depth and/or colorspace format may be used.

[0063] In order to activate or deactivate a stenciling feature of the invention, a user may provide a signal. The signal may be a mouse click, a switch toggle, a speech command, a button press, a key press, a touch-sensitive pad press, a button release, or any other indication performed by the user. In order to deactivate a stenciling feature after it has been activated, a user may repeat the signal provided for activation or may provide a different signal. Alternatively, the stencil may self-activate and/or self-deactivate.

[0064] A user may use painting and erasing features of a computer graphics application to create and/or edit an image. In a normal operation mode of a computer graphics application, the paint and erase strokes are blended into an image that is stored and/or displayed. Figure 1 is a block diagram 100 featuring a method of blending paint strokes 102 and erase strokes 104 into an image 106. For example, both paint strokes 102 and erase strokes 104 operate directly on a single image, which serves as both a stored image and a displayed image 106. The displayed image may simply be a graphical rendering of the stored image, as it appears on a CRT screen or other computer screen. Alternately, a separate image used for display may be maintained by copying a stored image into the displayed image at various frames of time during an editing process.

[0065] A user may switch from a normal operation mode to a stenciling mode by activating a stenciling feature. Upon deactivation of the stenciling feature, the normal operation mode of Figure 1 is resumed. Figure 2 illustrates a stenciling method of the invention, further described in the block diagram 300 of Figure 3.

[0066] In step 302 of Figure 3, the stenciling method creates a first texture based on user input provided either before or upon activation of the stenciling feature. The first texture is illustrated at reference 202 of Figure 2. The first texture 202 includes pixels with values indicating levels of protection to be applied to corresponding pixels of the protected image, shown at reference 204 of Figure 2. The user may select these pixels, for example, by using a graphical interface device to draw a boundary within which (or outside of which) to apply one or more protection levels. The user may specify a single protection level, multiple discrete protection levels, or ranges/gradients of protection levels for application in the selected region.

[0067] In step 304 of Figure 3, the stenciling method redirects graphical input from the stored/protected image into a second texture. Accordingly, Figure 2 shows graphical input 206 that is directed into a second texture 208. The graphical input 206 may represent one or more brush strokes, such as paint and/or erase strokes. The second texture accumulates graphical data from the user brush strokes performed throughout the time that the stenciling feature remains active. The protected image 204 is not modified as long as the stencil remains active and the second texture continues to accumulate graphical input.

[0068] Since the protected image 204 is not modified while the stencil is active, simply displaying the protected image 204 does not provide real-time feedback for the user.

This is because the protected image does not reflect the brush strokes performed by the user while the stencil is active. Thus, a display image is needed that is separate from the protected image 204. The stenciling method of Figure 3 creates a display image by first copying the protected image in step 306. Step 306 is illustrated in Figure 2 by arrow 210 extending from the protected image 204 to the display image 212. Then, step 308 of Figure 3 modifies the second texture values using the first texture. Step 308 is illustrated in Figure 2 by arrow 214 extending from the first texture 202 to the second texture 208. Step 308 may involve attenuating a color intensity value at a pixel of the second texture according to a value of the first texture corresponding to that pixel. This is described in more detail elsewhere herein. Step 310 of Figure 3 blends the modified second texture pixels into the display image by performing a compositing operation, for example, by performing an overlay operation in which pixel values copied from the protected image are combined with corresponding pixel values of the modified second texture. Steps 308 and 310 may be performed substantially simultaneously, as in a consolidated operation.

[0069] The display image can reflect real-time user brush strokes while the stencil remains active, as well as preserve protection level(s) specified by the first texture, regardless of the number of brush strokes performed. This is because, in a preferred embodiment, *each update* of the display image is performed by *re-copying* pixel values of the original protected image 204 into the display image pixels according to step 306, and then compositing the current modified second texture pixels with the display image pixels according to step 310. The display image is updated at a rate that provides sufficient real-time user feedback, for example, at rates of up to about 30 times per second or more. Finally, step 312 directs that, upon deactivation of the stencil by the

user, the second texture, modified by the first texture, is blended into the protected image 204. Step 312 is illustrated in Figure 2 by arrow 216 from the second texture 208 to the protected image 204.

[0070] Figures 4A and 4B demonstrate an implementation of the stenciling methods illustrated in Figures 2 and 3. Figure 4A is a schematic diagram 400 that shows an image resulting from the application of orange paint strokes 402 over a circular blue paint stroke 404 without a stencil. Figure 4B shows the image 420 wherein a stencil 426, or mask, having a 100% protection level is active during the application of orange paint strokes 422 over a previously-painted circular blue paint stroke 424. Thus, the orange paint strokes 422 applied following activation of the 100% stencil do not affect the stenciled region 426 illustrated in Figure 4B.

[0071] In terms of the steps of Figure 3 and component references of Figure 2, the implementation of the stenciling method in Figure 4B begins with a protected image 204 containing the original blue circle 424. Then, as shown in step 302 of Figure 3, a first texture is created by the user. In the implementation of Figure 4B, the user "lassos," or otherwise defines the region shown inside the irregular shaped, closed boundary at reference 426, and specifies that the level of protection to be applied for all points within this region is 100%. Thus, this region is completely shielded, or masked, from subsequent paint strokes as long as the stencil remains active. The stenciling method of Figure 3 is performed as the user applies the orange paint strokes 422 while viewing the display image 212. Finally, the user deactivates the stencil by providing a signal, such as a mouse click, a button release, or other signal. At this point, the display image, in a sense, becomes "permanent," since the modified second texture is blended into the

protected image after having accumulated all the brush strokes performed while the stencil has remained active, and after having been modified according to the protection levels of the first texture.

[0072] Each brush stroke that is accumulated in the second texture may be represented, at least in part, by a scratch texture. A scratch texture indicates one or more intensity values of pixels (and/or texels) associated with each brush stroke. A scratch texture may be considered to be a "template" describing characteristics of a brush used to perform a brush stroke. For example, the brush may be assigned a certain width, a varying width along the length of a stroke, a certain shape characteristic of the end of the brush, and/or a "falloff" or transition region along a portion of the brush, such as along the edges of the brush. The intensity values in the scratch texture may be used to determine the intensity of a selected color at a given pixel within the brush stroke. For example, a scratch texture comprising values representing a given brush stroke may represent a "falloff" region of the brush stroke by assigning intensity values from 1 (full intensity) at the center, or near to the center, of the stroke, down to 0 (no intensity) at points at the edges of the stroke and beyond. Thus, a blended composite of the scratch texture with a paint texture containing a single monochromatic color value would portray a paint stroke with the most intense color at the center of the stroke, falling off to no color (zero intensity) at the edges of the stroke. As each brush stroke is added to the second texture, a compositing operation is performed to combine the most recent brush stroke into the accumulated brush strokes. Each of these brush strokes may be based on scratch textures. For example, each brush stroke may simply be represented by its corresponding scratch

texture, where scratch texture intensity values are multiplied by a monochromatic color value.

[0073] Figure 5 is a schematic diagram 500 portraying pixel values of a scratch texture as grayscale intensity levels, where the scratch texture represents a paint stroke with a transition, or falloff, region. The paint stroke 500 is centered at point 0, and the falloff region begins at a distance f1 from the center and extends to a distance f2 from the center. Figure 6 shows a graph 600 depicting intensity level (y-axis, 602) of the scratch texture as a function of distance from the center 0 (x-axis, 604). Here, a linear falloff is depicted. The falloff region may, alternatively, be nonlinear.

[0074] Figures 7A and 7B show block diagrams 700, 720 that feature methods of blending graphical input 702 into any generic target image or texture 704 – both with and without using scratch textures. An example of the target texture 704 is the second texture 208 in the method illustrated in Figure 2. Graphical input 702 is based on the position of a graphical user interface at successive frames of time. An individual brush stroke is divided into a series of stroke segments that are separately blended into a scratch texture 706 as the stroke proceeds. The scratch texture 706 is then blended into the target image/texture 704.

[0075] Equations 1 through 5 show an illustrative blending algorithm used in the methods of the invention, where pixels are represented in RGBA format.

$$Ci = \text{overlay}(Ai, Bi): \tag{1}$$

$$Ci_r = Ai_r + (1-Ai_a)*Bi_r \tag{2}$$

$$Ci_g = Ai_g + (1-Ai_a)*Bi_g \tag{3}$$

$$Ci_b = Ai_b + (1-Ai_a)*Bi_b \tag{4}$$

$$Ci_a = Ai_a + (1-Ai_a)*Bi_a \quad (5)$$

where overlay(Ai, Bi) indicates an overlay compositing operation performed for corresponding pixels Ai and Bi of textures A and B using Equations 2-5; C is a composite of textures A and B; Ai_r, Ai_g, Ai_b, and Ai_a are the red, green, blue, and alpha channels of pixel Ai, respectively; Bi_r, Bi_g, Bi_b, and Bi_a are the red, green, blue, and alpha channels of pixel Bi, respectively; Ci_r, Ci_g, Ci_b, and Ci_a are the red, green, blue, and alpha channels of pixel Ci, respectively; and the results of computations are clamped to the range [0,1].

[0076] The overlay function in Equation 1 may be used, in effect, to assign color to pixels of a scratch texture. Thus, for each pixel location of a scratch texture containing a paint stroke, and for a given paint color P, methods of the invention may blend an attenuated paint color on top of a color pixel in a target texture according to Equation 6:

$$T \leftarrow \text{overlay}(i*P, T) \quad (6)$$

where i is the intensity associated with the pixel location in the scratch texture, T on the right side of Equation 6 represents the current color pixel in the target texture (prior to blending), and the "T ←" symbol represents writing the blended result into the target texture, replacing the current pixel value in the target texture.

[0077] The scratch texture may contain an erase stroke, and methods of the invention may blend an erase stroke into a target texture by attenuating existing target pixels according to Equation (7):

$$T \leftarrow (1-i)*T \quad (7)$$

[0078] The stenciling method illustrated in Figures 2 and 3 involves blending textures. For example, step 308 involves modifying values of the second texture 208 using the first texture 202, step 310 involves blending the second texture 208 into the display image

212, and step 312 involves blending the second texture 208 into the protected image 204. Moreover, the step of directing graphical input 206 into the second texture 208 may involve blending into the second texture 208 a scratch texture for each brush stroke performed while the stencil remains active.

[0079] In an embodiment of the invention where pixels are represented in RGBA format, the following is an example algorithm for performing the method illustrated in Figures 2 and 3 where the graphical input includes paint strokes each represented by a scratch texture. For each pixel location in the scratch texture:

(1) Blend the paint color P, attenuated according to the scratch texture, on top of the second texture at the current pixel location:

$$C \leftarrow \text{overlay}(i*P, C) \tag{8}$$

(2) Copy the protected image at the current pixel location into the display image:

$$D \leftarrow T \tag{9}$$

(3) Attenuate the second texture at the current pixel location according to the first texture, and blend it into the display image:

$$D \leftarrow \text{overlay}((1-s)*C, D) \tag{10}$$

where i is the intensity associated with the current pixel location in the scratch texture, s is the protection level associated with the current pixel location according to the first texture; C represents the pixel in the second texture corresponding to the current pixel location; T represents the pixel in the protected image corresponding to the current pixel location; and D represents the pixel in the display texture corresponding to the current pixel location.

[0080] A partial mask, or partial stencil, is applied in the case where the first texture contains protection level values less than 1 but greater than 0. The methods of Figures 2 and 3 allow a user to specify a first texture such that an initial version of a selected region of an image is maintained at a specified opacity in any subsequent composite, whether or not there are overlapping brush strokes applied while the stencil remains active. Various implementations of partial stenciling using the methods of Figures 2 and 3 are illustrated in Figures 8-10D.

[0081] Figure 8 is a schematic diagram 800 demonstrating the application of multiple, overlapping paint strokes in a region of an image protected by a method *other than* the method shown in Figures 2 and 3. In Figure 8, the partial mask simply attenuates the color intensity of each individual paint stroke according to a user-designated level, without the use of an accumulating stencil texture. Here, the attenuation level is 75%; thus, the intensity of a brush stroke applied within the protected region is diminished to 25% of the value it would have if applied outside the protected region. A blue rectangular brush stroke 802 is shown for contrast. Each of the brush strokes applied in Figure 8 has a size and shape as shown at reference 804, and each brush stroke is applied within a protected region of the image. Circle 806 shows the result of the application of one paint stroke following activation of the partial mask. Circle 808 shows the result of the application of two overlapping paint strokes following activation of the partial mask, and circle 810 shows the result of the application of three overlapping paint strokes following application of the partial mask. Circles 808 and 810 demonstrate the build up of color due to overlapping paint strokes. Although the partial mask has reduced the

intensity of each paint stroke, individually, it has not limited the aggregate effect of the multiple, overlapping paint strokes.

[0082] Figure 9 is a schematic diagram 900 that demonstrates how the stenciling method illustrated in Figures 2 and 3 prevents buildup of color caused by multiple, overlapping paint strokes within a partially-masked region. A blue rectangular paint stroke 802 is shown for contrast. Each of the paint strokes applied in Figure 9 has a size and shape as shown at reference 804, and each brush stroke is applied within a protected region of the image. The protection level is 75%; thus, the change in the partially protected region is limited to 25%, as specified in the first texture, regardless of how many brush strokes are applied while the stencil is active. Circle 906 shows the result of the application of one paint stroke following activation of the partial mask. Circle 908 shows the result of the application of two overlapping paint strokes following activation of the partial mask, and circle 910 shows the result of the application of three overlapping paint strokes following activation of the partial mask. The change in the partially protected region is limited to the maximum specified by the first texture. This maximum change is reached after application of a single paint stroke, shown in circle 906; therefore, there is no difference between the color intensities of circles 906, 908, and 910.

[0083] Figures 10A-10D further illustrate the effect of the application of partial stencils according to an embodiment of the invention. Figure 10A is a schematic diagram 1000 illustrating a circular paint stroke 1002 having a blue color value. Figure 10B is a schematic diagram 1010 illustrating the result of applying two overlapping orange paint strokes 1012, 1014, each at 50% intensity, onto the circular blue paint stroke 1002 without a stencil. The full intensity orange stroke 1016 is shown for reference. The

orange intensity in the region where the two orange strokes overlap 1018 exceeds 50% of the reference 1016, a result that is expected since the two strokes 1012, 1014 are blended without modification by a stencil.

[0084] Figure 10C is a schematic diagram 1020 illustrating the result of applying two overlapping orange paint strokes 1022, 1024, each at 50% intensity, onto the circular blue paint stroke 1002 following activation of a 50% protection stencil over the entire circular region. In Figure 10C, the orange intensity in the overlap region 1026 is higher than in the region outside the overlap, but does not exceed the maximum 50% of full intensity 1016 imposed by the stencil. Figure 10D is a schematic diagram 1030 illustrating the result of applying three overlapping orange paint strokes 1032, 1034, 1036, each at 50% full orange intensity, onto the circular blue brush stroke 1002 following activation of a 50% protection stencil over the entire circular region. As in Figure 10C, the region of overlap 1038 of the three orange paint strokes does not exceed the maximum 50% of full intensity 1016 limit imposed by the stencil. In this way, Figures 10A-10D demonstrate that the invention may be used to partially stencil an object while still allowing overlapping paint strokes to build in intensity, up to a specified maximum.

[0085] The graphical input directed in the second texture following activation of a stencil may contain erase strokes. Real-time display of erase strokes requires modification of paint strokes applied both before and after activation of the stencil. Since the second texture contains data from paint strokes applied after activation of the stencil, but not before, a method of the invention includes the step of modifying the pixels of the protected image prior to copying pixels of the protected image into the display image and compositing the modified second texture pixels with the display image pixels.

[0086] Accordingly, Figure 11 is a schematic diagram 1100 illustrating a method for blending graphical input 1102 into an image 204 while a full or partial stencil is active, where the graphical input 1102 includes erase strokes. The method illustrated in Figure 11 is further described in the block diagram 1200 of Figure 12.

[0087] In step 302 of Figure 12, the method creates a first texture based on user input. The first texture is illustrated at reference 202 of Figure 11. In step 1202 of Figure 12, the stenciling method redirects graphical input, including one or more erase strokes, into a second texture 208. Accordingly, Figure 11 shows graphical input 1102 that is directed into the second texture 208. Unlike the method described in Figures 2 and 3, the protected image 204 in Figure 11 is modified by the erase strokes while the stencil is still active. This is shown in step 1204 of Figure 12 and is indicated by arrow 1104 in Figure 11. However, each erase stroke is filtered by the first texture before it modifies the protected image 204, so that the protection of stenciled regions is maintained. This is indicated by arrow 1106 in Figure 11. The remainder of the process is as described in conjunction with Figures 2 and 3, following steps 306, 308, 310, and 312 of Figure 12.

[0088] In an embodiment of the invention where pixels are represented in RGBA format, the following is an example algorithm for performing the method illustrated in Figures 11 and 12 for an erase stroke represented by a scratch texture. For each pixel location in the scratch texture:

(1) Attenuate C at the current pixel location according to the scratch texture:

$$C \leftarrow (1-i)*C \tag{11}$$

(2) If the alpha value of the protected image at the current pixel location is greater than the protection level at the current pixel location specified by the first

texture (If Ta > s), attenuate the protected image at the current pixel location down to a minimum of s:

$$T \leftarrow \max(1-i, s/Ta)*T \quad (12)$$

(3) Attenuate the second texture at the current pixel location according to the first texture, and blend it into the display image:

$$D \leftarrow \text{overlay}((1-s)*C, D) \quad (13)$$

where i is the intensity associated with the current pixel location in the scratch texture, s is the protection level associated with the current pixel location according to the first texture; C represents the pixel in the second texture corresponding to the current pixel location; T represents the pixel in the protected image corresponding to the current pixel location; and D represents the pixel in the display texture corresponding to the current pixel location.

[0089] Various implementations of partial stenciling during the application of erase strokes using the methods of Figures 11 and 12 are illustrated in Figures 13-15. Figure 13 is a schematic diagram demonstrating a removal of color by multiple, overlapping erase strokes within a region of an image protected by a method *other than* the method shown in Figures 11 and 12. In Figure 13, the mask simply scales each erase stroke according to a user-designated level. Here, the "protection" level is 75%, thus, the effect of each erase stroke is reduced by 75% before application. A blue rectangular brush stroke 802 is shown for contrast. Each of the erase strokes applied in Figure 13 has a size and shape as shown at reference 804, and each erase stroke is applied within a protected region of the image. Circle 1302 shows the result of the application of one erase stroke over the reference 804, following activation of the partial mask. Circle 1304 shows the

result of the application of two erase strokes over the reference 804, following activation of the partial mask. Circle 1306 shows the result of the application of three erase strokes over the reference 804, following activation of the partial mask. With each successive erase stroke, the color is further attenuated. Although the partial mask has limited the effect of each erase stroke individually, it has not limited the aggregate effect of multiple, overlapping erase strokes.

[0090] Figure 14 is a schematic diagram 1400 that demonstrates how the stenciling method illustrated in Figures 11 and 12 prevents removal of color due to multiple overlapping erase strokes within a partially-masked region. Each erase stroke is applied within a protected region. The protection level specified by the first texture over the protected region is 75%. Circle 1402 shows the result of the application of one erase stroke over the reference 804 following activation of the partial mask. Circle 1404 shows the result of the application of two erase strokes over the reference 804 following application of the partial mask, and circle 1406 shows the result of the application of three erase strokes over the reference 804 following application of the partial mask. Here, the attenuation is limited to the amount specified by the partial mask. Multiple erase strokes do not cause the attenuation to exceed the amount specified by the partial mask. As in the case of painting, this result serves the needs of users who wish to use a partial mask to specify a maximum extent to which an image is allowed to change in selected, stenciled regions.

[0091] Figure 15 is a schematic diagram 1500 illustrating the result of applying two overlapping erase strokes 1502, 1504, each at full intensity, onto a circular blue paint stroke 1002 following activation of a 50% stencil over the entire circular region. The

intensity of the underlying blue paint stroke 1002 is not attenuated below the 50% limit imposed by the stencil, even in the overlapping region 1506, despite the fact that two erase strokes have been applied to this region.

[0092] A partial mask may be used along the edges of a full mask in order to provide a smooth transition between masked and unmasked portions of a stenciled image. Figure 16A is a schematic diagram 1600 illustrating the application of a single paint stroke (100% black) across an image following activation of a partial mask. The partial mask is applied as a feathering function along the edges of a full mask, indicated by the red circle 1602, *not* in accordance with the partial stenciling methods of Figures 2 and 3. As discussed with respect to Figure 8, the partial mask shown attenuates the color intensity of each individual paint stroke according to a user-designated level without the use of an accumulating stencil texture. Figure 16B is a schematic diagram 1610 illustrating the buildup of color within the partially protected region following application of three overlapping paint strokes (100% black) across the entire image, and Figure 16C 1620 illustrates the buildup of color within the partially protected region following application of five overlapping paint strokes (100% black) across the entire image. In contrast to the method above, applying a partial stencil according to the method of Figures 2 and 3 results in no buildup of color within a partially protected region following application of multiple overlapping paint strokes within the partially protected region.

[0093] According to an embodiment of the invention, an individual brush stroke is divided into a series of stroke segments that are separately blended into a scratch texture. The invention provides a method that prevents double-blending of overlapping portions

of segments of the individual brush stroke, thereby avoiding undesired artifacts at the segment ends.

[0094] Figure 17 is a schematic diagram 1700 showing two contiguous pillbox segments 1702, 1704 of a brush stroke. Figure 17 shows the two segments of the brush stroke both with 1706 and without 1708 a double-blending artifact in the overlapping portion. Each segment of the brush stroke is represented using a pillbox shape. Brush stroke 1706 is divided into two segments 1702 and 1704, and brush stroke 1708 is divided into two segments 1710 and 1712. The overlapping portion of the two segments in brush stroke 1706 results in higher intensities in the circular region 1714. The blending method performed for brush stroke 1708 avoids the higher intensities in the region 1716 where the ends of the pillboxes overlap, resulting in a brush stroke of even intensity.

[0095] The blending method uses a scratch texture to describe the size and shape of the brush. By assigning a new pixel value to the scratch texture only if it exceeds the existing value at that pixel, the double-blending artifact is avoided. Figure 18 is a block diagram 1800 featuring a method of blending a scratch texture containing a single brush stroke into a target image or texture. Step 1802 indicates the performance of a single brush stroke by a user. In step 1804, candidate pixel values are generated corresponding to the brush stroke. In step 1806 it is determined for a candidate pixel value whether the candidate value is greater than the existing pixel value in the target image/texture. If so, step 1808 writes the candidate pixel value into the pixel of the scratch texture, and step 1810 and 1811 proceed through the remaining candidate pixels. The blending step 1812

is performed substantially upon completion of the performance of the paint stroke by the user.

[0096] A computer hardware apparatus may be used in carrying out any of the methods described herein. The apparatus may include, for example, a general purpose computer, an embedded computer, a laptop or desktop computer, or any other type of computer that is capable of running software, issuing suitable control commands, receiving graphical user input, and recording information. The computer typically includes one or more central processing units for executing the instructions contained in software code that embraces one or more of the methods described herein. The software may include one or more modules recorded on machine-readable media, where the term machine-readable media encompasses software, hardwired logic, firmware, object code, and the like. Additionally, communication buses and I/O ports may be provided to link any or all of the hardware components together and permit communication with other computers and computer networks, including the internet, as desired.

Equivalents

[0097] While the invention has been particularly shown and described with reference to specific preferred embodiments, it should be understood by those skilled in the art that various changes in form and detail may be made therein without departing from the spirit and scope of the invention as defined by the appended claims.

[0098] What is claimed is: